2.1 Part 1: Viterbi Decoding

I implemented the iterative recursion inference and backtracking described in course matrials and the homework handout. The implementation is based on matrix computation.
Running time:

```
Data reading and training took 3.930430 seconds
```

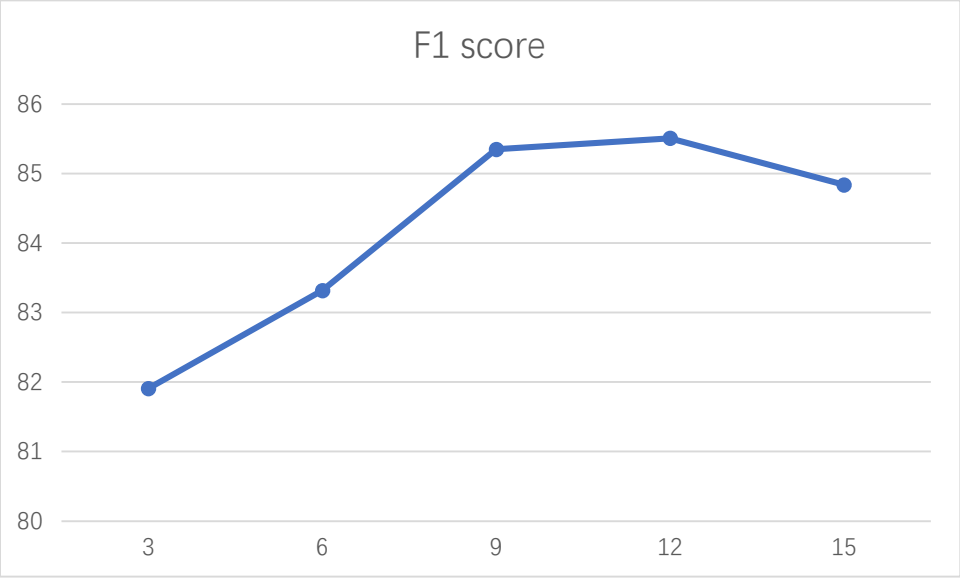
Results:

```
Labeled F1: 75.72, precision: 4218/5198 = 81.15, recall: 4218/5943 = 70.97
```

2.2 Part 2: CRF Training

1. For the emssion features, I use the given emission features.
2. For the transition weights, I modify the weights from the HMM model to meet the hard constraints that only B-X or I-X can transition to I-X. For a "to" entry I-X, any weight whose "from" entry is not B-X or I-X is set to -1000. The weights are attached as "transition.csv".
3. I implemented Adam as my optimizer.
4. A single epoch takes 144 seconds, as shown in the training log "NER_train.o5525641".
5. I trained the CRF model for 15 epochs using 1.03 hours, and attach my training log "NER_train.o5525715".

After every 3 training epochs, I save the model parameters and test it on the development set. Here is the F1 score chart:



You can see that after the 12th epoch, my model achieves the best performance on the development set, as shown below:

```
Labeled F1: 85.51, precision: 5069/5913 = 85.73, recall: 5069/5943 = 85.29
```

Thus, I decide to use the model the 12th epoch as the final CRF model. The parameters are

attached as “model_12.npy”. The CRF's output on the testb blind test set in CoNLL format is attached.